Jack Dohany
627 Vera Ave
Redwood City, CA 94Ø61

415-367-7781

July 29, 1995

Rod Gowan
RMG ENTERPRISES
14784 South Quail Grove Circle
Oregon City, OR 97Ø45

Hi Rod,

Thanks for your check #6712 for the dual eproms.

Work on the hard drive is proceeding slowly. I too wish it could be done sooner. If someone wants to pay me $2Ø per hour to work on it 6Ø hours per week, then it could be done in a month. Otherwise, since I need to eat and pay bills, I'll have to continue working on it in my spare time. I am, by the way, getting help from some former-Timex now-IBM experts: Bob Orrfelt, Terry Greenlee, Pat Morrissey et al.

Well, I finally got all the clock software and documentation just the way I want it. Since the Smartwatch chip is the easiest and cheapest way to go, I hope it will become the "standard" 2Ø68/Spectrum clock/calendar device. With that in mind, I wanted to make the software as good as it could be.

The enclosed Smartwatch chip is a gift. All clock software that I've written is in the public domain. You can distribute the software and documentation however you want. I would prefer not to be involved in the distribution process; I'd rather spend my spare time on the hard drive project.

If you want to order Smartwatch chips from Jameco or wherever, and sell them with the software, go right ahead. But I'd suggest just distributing the software and documentation, and letting folks buy or find their own Smartwatch chips.

If you would like to have copies of the technical data on the chip and software, which is many pages long, let me know. It would mainly be of interest to machine-code experts.

Best regards,

Jack

Jack Dohany
627 Vera Ave
Redwood City, CA 94Ø61

415-367-7781

July 1995

*** A CLOCK/CALENDAR FOR 2Ø68/SPECTRUM ***

There's a song that goes: "Does anybody really know what time it is? Does anybody really care?" Well, if you care, and if you want your computer to know the time (and date) then read on.

When I first bought a Timex computer, I expected it to have a built-in clock/calendar chip, and was shocked to find that it didn't. After all, the Timex folks specialize in digital watches, don't they?

Anyhow, I have finally remedied the situation. The IBM XT/AT clones used a clock/calendar chip called the Dallas Smartwatch, also known as a No-Slot Clock, because it was designed to fit in almost any rom or eprom socket, piggybacking any existing rom or eprom. The chip is now obsolete, along with the XT and AT, but the chip is still available, and can be easily installed in the 2Ø68. No soldering or wiring changes are needed.

This document briefly describes the chip, and tells you where to get it, how to install it, and how to use the software I've written for it. This document does not give technical info about the chip or the software; if you need such information, or if the software doesn't suit your needs, please contact me or your supplier.

NOTE that I do NOT sell the chips. In fact, I don't even sell the software! I have placed it in the public domain. Whoever distributes the software may of course charge for that service, and may also sell the chips. You are welcome to send me a donation for the work of writing and sharing the software, but this is neither required nor expected.

CHIP DESCRIPTION

The Dallas Semiconductor DS1216E Smartwatch chip has 28 pins on the underside, and 28 pin receptacles on the topside. It contains a built-in oscillator and two built-in lithium batteries which should last more than ten years. It may seem expensive compared to other clock/calendar chips, but when you consider that other chips require a hand-wired circuit board, a battery socket and a battery, then you see that the Smartwatch chip is quite economical.

WHERE TO GET THE CLOCK CHIP

The DS1216E Smartwatch is avaiable from:
JAMECO ELECTRONICS, 1-8ØØ-831-4242.

On page 19 of the May-July 1995 JAMECO catalog, the chip has part no. 111Ø8 and costs $15.95.

The chip is also available from JDR Microdevices. You may also find one at your local computer store, or at a computer swap meet. You can also steal one from an old defunct XT/AT clone.

1. Disconnect all cables and interfaces from the 2Ø68.
2. Remove the upper half of the case. CAREFULLY unplug the keyboard cable from the main board.
3. Remove the EXROM from its socket. Note: notch faces rear.
4. Insert the Smartwatch chip in the EXROM socket, with its notch facing rear. Use CMOS handling precautions.
5. Insert the EXROM in Smartwatch socket, notch facing rear.
6. Using a pair of pliers or other tools, remove the ribs in the upper half of the case above the EXROM. This provides the additional 3/8" of vertical clearance required by the Smartwatch chip.
7. Fasten a piece of friction tape or masking tape on the left side of the EXROM, to prevent its pins from being contacted by the metal cartridge door mechanism.
8. Re-assemble the computer, and try out the software. Begin with SETCLK, which turns the chip's oscillator on etc.

NOTE: If you don't trust yourself to perform the above procedure, then find someone whom you do trust. I will be happy to install the chip for you free of charge; it takes me about a minute. You would need to send me your computer, the chip, and return postage. If you have only one 2Ø68, I could send you a loaner to use while yours is gone.

BRIEF SOFTWARE DESCRIPTION

The clock software consists of one Mscript text file (this document), one machine-code program, and two BASIC programs, each containing a relocatable machine-code program in a Line Ø REM statement. The programs all work on a T/S 2Ø68 with or without Spectrum emulation. The software may be distributed on cassette or disk. No changes are needed to the software for different mass storage systems. File names may vary.

NAME	TYPE	START	LENGTH	COMMENTS
CLKDOC	CODE	46927	n/a	This text file, Mscript format
SETCLK	CODE	64ØØØ	9ØØ	Non-relocatable. Turns clock on or off. Sets day, date, mode and time. Needs no BASIC.
CLOCK1	BASIC	n/a	n/a	BASIC kernel containing DCLOCK in a Ø REM statement. May be merged with other BASIC programs, or used as the starting point for writing a clock-related program.
CLOCK2	BASIC	n/a	n/a	BASIC kernel similar to CLOCK1, but containing RCLOCK instead of DCLOCK.

Usage of the clock software is described on the next page. You should make and use backup copies.

NOTE: loading examples are given using cassette commands. Commands to be used with your disk system will vary.

Reset computer, then load SETCLK thus:

```
CLEAR 63999              The program will occupy
LOAD "SETCLK" CODE       locations 64000 to 64899.
```

To make a backup: SAVE "SETCLK" CODE 64000,900

Then use the program as follows:

```
RANDOMIZE USR 64000
```

You'll see the day, date and time, and this menu:

```
0=EXIT  1=DAY   2=DATE

3=TIME  4=MODE  5=AMPM  8=ON/OFF
```

Note: Before changing the date, note the date displayed. It may be an indication of how old the chip (and its batteries) are. Like the odometer on an old car, this is not reliable info.

Press KEY 1 to change the day of the week (Sunday to Saturday). The clock chip does not do this automatically based on the date. It merely advances the day counter after midnight.

Press KEY 2 to enter a new date. Example: 072695 or 07-26-95.

NOTE: When entering time or date, there is no need to press ENTER. Just type all 6 digits (with or without separators). If you press ENTER before 6 digits are typed, the entry will be cancelled, and the time or date will be left unchanged.

NOTE: If you prefer 26.07.95 (European style), check out CLOCK2, which lets you display clock data in a variety of ways.

Press KEY 3 to change mode, if desired, before setting time. In 12-hour mode, 2 PM would be shown as 02:00:00 PM. In 24-hour mode, 2 PM would be shown as 14:00:00.

After changing modes, time may be incorrect. Correct it as follows:

Press KEY 4 to enter the time. Example: 031400 or 03:14:00.

In 12-hour mode, press KEY 5 to change the AM/PM indicator if necessary. In 24-hour mode, KEY 5 has no effect.

Press KEY 8 to turn the chip's oscillator off or on, if desired. It should only be left off if you are testing a chip and plan to remove and store it. The chip uses slightly less battery power when the oscillator is off.

Press KEY 0 to exit the program and return to BASIC.

Ideally, you should only need to use SETCLK once, unless you move to a different time zone or wish to change the mode. However, no clock keeps time perfectly forever. If the time drifts after a while, you can use SETCLK to correct it.

These two very short BASIC programs serve as "kernels", that is, they can serve as a starting point for writing your own BASIC program that uses day/date/time, or they could be merged with existing BASIC programs.

Both CLOCK1 and CLOCK2 contain a short machine-code program that reads the Smartwatch chip data. Each program reports the name, location and length of the machine code, which is contained in a REM statement at Line Ø. The machine code is relocatable and Spectrum-compatible.

The machine code programs (called DCLOCK and RCLOCK) could be saved separately and later reloaded for use in a different area of memory, such as 65ØØØ up. This would seldom be necessary.

CLOCK1
The DCLOCK program included with CLOCK1 is very simple: it sends the clock data in printable form to the current output channel. The way to use it, assuming that the variable DCLOCK holds the address of the program, is something like this:

```
PRINT AT 1Ø,Ø;: RANDOMIZE USR DCLOCK
```

You could print the clock data on paper thus:

```
LPRINT;: RANDOMIZE USR DCLOCK
```

Clock data will appear something like this:

```
Wednesday Ø7-26-95   Ø3:31:Ø4 PM
```

Note that DCLOCK doesn't report hundredths of a second, since on the screen they would appear as a blur. If hundredths of a second are needed, CLOCK2 should be used.

Incidentally, the DCLOCK code is 325 bytes long.

CLOCK2
The RCLOCK program included with CLOCK2 is somewhat more complex and versatile. It reads the clock data into the first 32 positions of an existing one-dimensional string array whose length is at least 32. You (or your program) must tell the program WHICH string array (A$ to Z$) to use, by poking location (RCLOCK+2) with the code of a letter from A to Z.

The CLOCK2 BASIC program finds the location of RCLOCK, creates a string array T$(32), tells RCLOCK to use T$, calls RCLOCK, and displays the clock data in T$ in various ways.

NOTE: If the array to be used by RCLOCK is non-existent or too short or multi-dimensional, RCLOCK will simply do nothing. It won't crash.

Incidentally, the RCLOCK code is 44Ø bytes long.

ARRAY LAYOUT
The RCLOCK program places clock data into the first 32 array locations thus:

1-9	Day of the week	*	2Ø-21	Hour (ØØ-23 or Ø1-12)
1Ø	space		22	: (colon)
11-12	Month (Ø1-12)		23-24	Minute (ØØ-59)
13	- (hyphen)		25	: (colon)
14-15	Date (Ø1-31)		26-27	Second (ØØ-59)
16	- (hyphen)		28	. (decimal point)
17-18	Year (ØØ-99)		29-3Ø	Second/1ØØ (ØØ-99)
19	space	*	31-32	spaces or AM or PM

(* = mode-dependent)

DAY OF THE WEEK FEATURE
There may be times when you or your program want the day of the week as a NUMBER from 1 to 7. You could write a BASIC routine to convert the array data to a number, but a simpler way is this: LET DAY=USR DCLOCK. This reads the clock data into the array in the same way as RANDOMIZE USR DCLOCK would. It also places the day of the week into numeric variable DAY as a number from 1 to 7. Any legal numeric variable name may be used. Note: Sunday=1, Monday=2, etc.

CONCLUSION
Please study CLOCK1 and CLOCK2 carefully. To LIST either program, use LIST 1 rather than LIST, since the use of LIST will cause the quite horrible listing of the machine code in Line Ø.

I believe that most or all of your clock-related needs can be met by the use of CLOCK1 or CLOCK2. If you have needs which are NOT met by either program, please let me know.

QUESTIONS AND ANSWERS

WHAT ABOUT LEAP YEAR?
The clock chip handles it automatically.

WHY CAN THE TIME BE SET AT 34:62:73?
Neither the clock chip nor the SETCLK program are foolproof. If you say that's what time it is, then so be it.

WHAT IF THE BATTERIES DIE?
When the clock batteries die, your clock chip will no longer keep time. The batteries are imbedded in epoxy and are not replaceable. So do you chuck the chip? Of course not! You send it to me, with $5. I'll disconnect the internal batteries and connect a small external lithium cell that'll last another ten years. I'll then return your chip.

WHAT IF SOMETHING IMPORTANT IS NOT DISCUSSED?
Well, I think I've covered everything important. If you do have any questions, just contact me. So long for now, and Happy Timekeeping!